Technical Note: Context Bandwidth and Local Predictors in Atlas Neural Networks

Central Observation

A context-driven neural network consists of two interacting computational systems:

- a **context system**, which determines the computational state of the network,
- a collection of **local predictors**, which perform the computation associated with that state.

The predictor does not operate in isolation. It always operates within a context.

Separation of Responsibilities

The context system answers the question:

> **"Where in the computational state space am I?"**

The local predictor answers:

> **"Given this computational state, what computation should be performed?"**

Learning is therefore the joint optimization of context and computation.

Context Bandwidth as a Design Parameter

Atlas neural networks expose context bandwidth as an explicit architectural parameter.

The designer may choose:

- low context bandwidth,
- medium context bandwidth,
- very high context bandwidth,

for each layer independently.

This choice is largely independent of the complexity of the local predictors.

The architecture therefore separates two independent design questions:

- How much contextual information should each layer receive?
- How powerful should each local predictor be?

This separation is largely absent in conventional ReLU networks, where the effective context representation emerges from the network architecture rather than being explicitly designed.

Context-Limited and Context-Saturated Layers

A useful distinction is between two operating regimes.

Context-limited

The predictor receives only a restricted description of the current input.

Learning must compensate for missing contextual information.

Increasing predictor complexity may yield diminishing returns because the predictor simply lacks sufficient context.

Context-saturated

The predictor receives an information-rich context representation.

Learning becomes primarily an organizational problem.

The challenge is no longer obtaining context, but discovering useful computational structure within it.

This shifts the computational bottleneck from context acquisition to predictor organization.

The Role of Local Predictors

A local predictor should not be viewed as memorizing individual contexts.

Instead, it learns reusable computational structure.

Its role is to:

- identify informative context variables,
- ignore irrelevant context,
- discover interactions between context variables,
- exploit statistical regularities,
- organize computation efficiently.

Backpropagation gradually improves this organization over training.

Factorization Through Context

The context does not directly compute the output.

Instead, it enables the predictor to factorize the mapping into reusable computational components.

Rather than learning one enormous mapping covering every possible situation, the predictor learns computations that are useful across many related contexts.

The context determines which portions of this shared computational structure become relevant for the current input.

In this sense, context acts as an organizational coordinate system for computation.

A Geometric Interpretation

The context representation partitions the input manifold into overlapping computational regions.

Within each region, the local predictor learns an approximation appropriate for that portion of the space.

Increasing context bandwidth generally produces a finer partition of the computational state space.

Increasing predictor complexity generally produces a more accurate approximation within each region.

These two forms of complexity are complementary rather than interchangeable.

Interaction Between Layers

Each layer receives both:

- the evolving feature representation,
- the contextual description.

Backpropagation coordinates all layers simultaneously.

Rather than each layer independently discovering a complete solution, the network develops a hierarchy of local predictors that progressively organize computation while operating under shared contextual information.

The interaction between predictors becomes as important as the predictors themselves.

Experimental Questions

Viewing Atlas networks in this way suggests several experimental directions.

Hold predictor complexity fixed while varying context bandwidth:

- How much bandwidth is required before performance saturates?
- Does optimal bandwidth increase with problem complexity?

Hold context bandwidth fixed while varying predictor complexity:

- When do larger predictors cease to improve performance?
- Does context richness reduce the required predictor size?

Vary both simultaneously:

- Is there an optimal trade-off between context bandwidth and predictor complexity?
- Can predictor capacity be exchanged for richer context?

Measure the context itself:

- entropy,
- redundancy,
- utilization,
- stability,
- frequency of occurrence,
- information carried per context bit.

These measurements may provide architectural diagnostics that are unavailable in conventional neural networks.

Towards a Mathematical Framework

This viewpoint suggests describing a context-driven architecture by two coupled functions:

- a context function that maps inputs into computational states,
- a predictor function that maps computational states into outputs.

Learning then becomes the joint optimization of both.

Rather than treating the network as a single nonlinear function, it may be more productive to study the interaction between context generation and context-dependent local prediction.

This decomposition provides a framework for analysing capacity, specialization, parameter sharing, and generalization in a unified way.

Summary

Context bandwidth provides a new architectural axis alongside depth, width, and parameter count.

In Atlas neural networks it becomes an explicit design parameter that can be independently controlled. The resulting architecture separates the problems of context generation and local prediction, allowing researchers to study their interaction directly.

From this perspective, the primary role of context is not to compute the output, but to organize computation. The primary role of the local predictors is not to memorize individual contexts, but to

discover reusable computational structure within an information-rich context space. Learning emerges from the continual interaction of these two systems under the guidance of backpropagation.

Take Away

The context provides the coordinates in computational state space, while the local predictor learns a factorized computation over that state space.